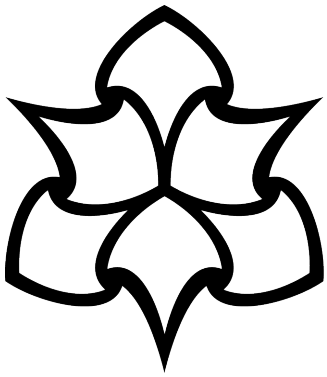


Dynamic Container-based Resource Management Framework of Spark Ecosystem

Nawab Muhammad Faseeh Qureshi[*α], Isma Farah Siddiqui[β], Asad Abbas[γ], Ali Kashif Bashir[σ] , Keehyun Choi[α], Jaehyoun Kim[α] and Dong Ryeol Shin[α]

[α]Sungkyunkwan University, South Korea

[β]Mehran University of Engineering & Technology, Pakistan

[γ]University of Lahore, Pakistan

[σ]University of Faroe Islands, Faroe Islands

[*]Corresponding Author Email: faseeh@skku.edu

***Abstract*— Apache Spark is known for its robustness in processing large-scale datasets in a distributed computing environment. This form of efficiency is highly observing because of the direct use of Random-Access Memory (RAM) in processing its resilient distributed datasets across the ecosystem. Recently, it is observed that, the memory utilization in computing spark jobs is mainly dependent on job containers, which are closely associated to persistent storage media components. Thus, spark jobs processing relevancy is tightly coupled to the type of storage container and in case of any dynamic resource allocation, the job loses its ratio of resource computation in existing container and increases a functional issue of processing large-scale datasets in spark ecosystem. In this paper, we propose dynamic container-based resource management framework, that shifts coupled associations of job profiles to dynamically available resource containers. Also, it relieves static container allocations and presumes them as a fresh piece of resource allocation for new job profile. The experimental evaluation shows that the proposed dynamic framework reduces wastage of resource allocations and increase ecosystem performance than default job profile in spark ecosystem.**

***Keywords*— Apache Spark, Storage containers, Job Coupling, Resource allocation and Management**

I. INTRODUCTION

Apache Spark is a data block analytical ecosystem, that processes large-scale datasets using various sub-components such as Core, SQL, Streaming, Machine Learning Library (MLlib), GraphX and SparkR [1]. It is a robust prototype with a sophisticated cluster computing environment, that facilitates users with high-level rich library of Java, Scala, Python and R languages [2]. Spark ecosystem enhances the existing functionalities of Apache Hadoop [3] and speeds up large-scale online data processing, run-time specialized query optimization methods, advanced data training and learning procedures and core-enabled parallel graph computations [4].

Apache Spark processes data blocks in the form of Resilient Distributed Dataset (RDD), that collects computation objects on inter-connected nodes of the cluster [5]. The purpose of generating an RDD is to store intermediate cluster results into distributed memory (RAM) against persistent storage such as DISK and SSD [6]. Thus, when a Spark job is scheduled to generate RDDs, the stage attribute divides the job into two phases, such as (i) ShuffleMapStage and (ii) ResultStage to format a processing route of executing tasks. The cluster manager then divides ShuffleMapStage between available network nodes and initializes executor paradigm to process the tasks. The ResultStage collects the calculated tasks result and store the files inside persistent containers [7] as shown in Figure-1.

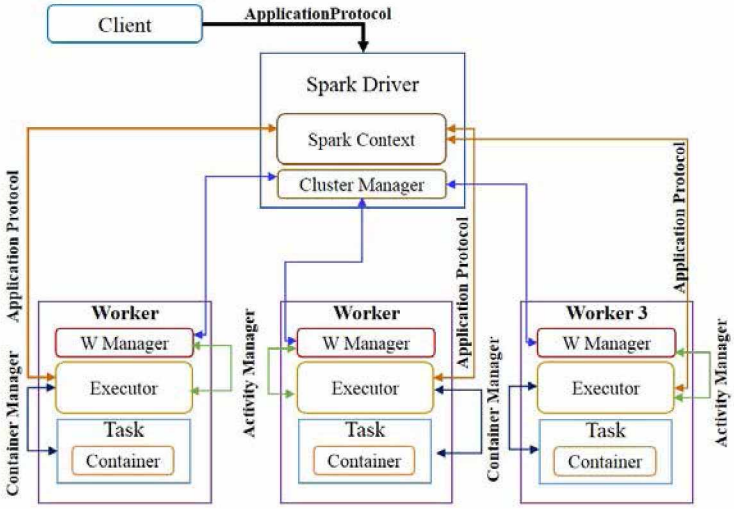


Figure 1. Default Apache Spark RDD Processing Architecture

The role of in-memory computation is limited only to the ShuffleMapStage and therefore, cluster nodes store first phase results into the distributed memory (RAM) of a worker [8].

Recently, it is observed that, the persistent storage that stores finalized RDD is tightly coupled to the matching types of the job containers, such as SSD cache memory to SSD persistent storage. As a result, when a random job is assigned to an uncatalogued group of cluster nodes, the tasks drop its assigned ratio of resource computation. For Example, if a job is assigned 22% computing resources for SSD, after getting the RDD from first stage of ShuffleMapStage, it is redirected towards the second phase of ResultStage belong to SSD container, however, if the cluster is not declaring a catalogue of persistent storage devices, the job drops its ratio of computing resources and outcomes with a fall of job profile

coupling with low performance in the cluster. This paper addresses a dynamic container-based resource management framework, that assigns job coupling profile to any immediate available resource container. Also, it flushes the catalogue with an update so the job arriving next the current job could not have a reason of resource wastage. The main contributions of the paper are given as:

1) RDD coupling identifier
2) Enhanced job profiler

The rest of the paper is divided into four sections. Section II briefs about related work of available resource management workflow. Section III describes the proposed methodology of the dynamic container-based resource management. Section IV evaluates our scheme with available cluster experimentation. Section V concludes the proposed approach with brief credentials.

II. Related Work

The container resource management is a built-in function, that was programmed by the Apache Hadoop developers [9] to provide fine-grained resource sharing for processing jobs. This initiative strengthened the concept of using multiple persistent storage devices in an Apache Hadoop cluster that was followed with an abstract design of heterogeneity-aware tiered storage perspective [10]. After this proposition, the issue of using multiple storage devices appeared and was resolved with an approach of storage-tier-aware data placement strategy [11]. With this enhancement, a combined approach of Apache Hadoop and Spark was introduced for processing datasets onto multiple storage devices [12] and introduced another approach of balancing big data processing storage activities with dynamic data placement [13]. After that, an independent experiment is conducted to evaluate the solo performance of Apache Spark in handling heterogeneous environment [14].

The dynamic container-based resource management is still yet to be discussed and through our proposed approach, the job profile that includes ResultStage storage objects would be placing resultant data to job coupled persistent storage.

III. Dynamic Container-Based Resource Management (DCRM)

The dynamic container-based resource management (DCRM) framework strengthens Spark job profile by applying associated coupling attributes to ResultStage and stores resultant data to the dynamically vacuum persistent storage as shown in Figure-2. This coupling is carried out in a systematic hierarchy with additional components such as (i) RDD Coupling Identifier and (ii) Enhanced Job Profiling.

A. RDD Coupling Identifier

By default, the Resilient Distributed Dataset (RDD) facilitates tasks with read, write, consistency, fault-recovery and straggler mitigation techniques. In order to couple dynamic resources allocation to the said functions, DCRM enhances the RDD structure with an attribute of dynamic identifier, that fetches the current state of the worker node and associate on-ground available persistent storage entities such as adding new entry of accessible storage media in the job as shown in Figure-3.

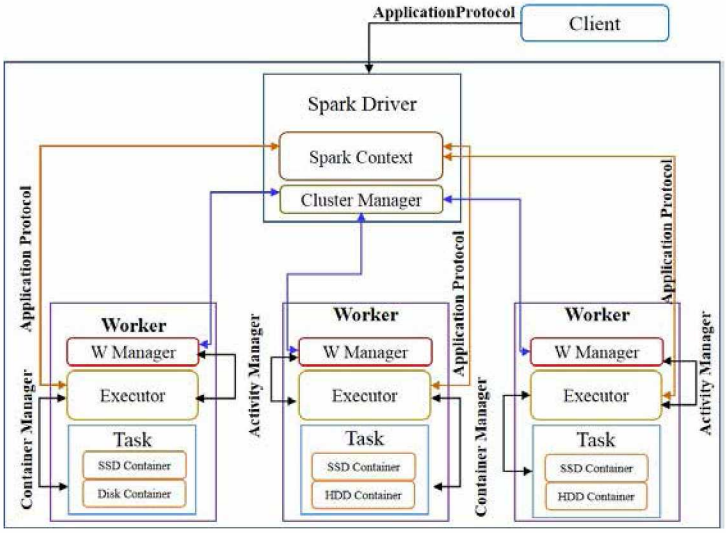


Figure 2. Dynamic Container-Based Resource Management Framework

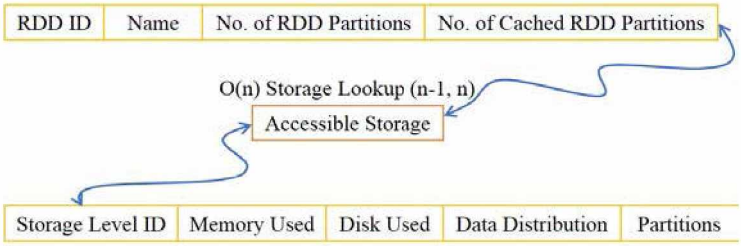


Figure 3. RDD Coupling Identifier

Thus, the formation of an enhanced RDD in terms of all the three creation ways such as parallelized collection, external datasets and generating a dataset from existing RDD could be obtained as,

$$Parallelized\ Collection_E = \sum_{Master\ i}^{Worker\ a \in Z} \left[Tokens, \left(\frac{Parallelized * O(n)Storage}{SortByKey} \right) \right] \quad (1)$$

$$External\ Dataset_E = \sum_{Master\ i}^{Worker\ a \in Z} \left[CSV(String), \left(\frac{SessionRead * O(n)Storage}{SortByKey(Path)} \right) \right] \quad (2)$$

$$DatasetE(RDD)_E = \sum_{Master\ i}^{Worker\ a \in Z} \left[Tokens(RDD), \left(\frac{Parallelized * O(n)Storage}{SortByKey(RDD)} \right) \right] \quad (3)$$

Where, '*tokens*' are the number of sequences followed by a distributed execution of a task, '*csv*' is an already available external dataset, '*parallelized*' function returns formation of distributed ShuffleMapStage tasks and '*sessionread*' is number of read count of rows in the external dataset.

B. Enhanced Job Profiler

The job profiler is a time barred function, that handles executions of RDD tasks in distributed worker environment. DCRM enhances its layered workflow with an addition of '*onStorageLookup*' strategy that manages tasks execution in between '*onJobStart*' and '*onJobEnd*' in such a way that, job

returns available lookup state of persistent storage entries and uses them in synchronizing parallel operations of the worker nodes in a distributed computing environment. This '*onStorageLookup*' is further categorized into three sub-functions such as '*onJSLookup*', '*onBGLookup*' and '*onSCLookup*'. The '*onJSLookup*' strategy delivers a stack of persistent log entries upon stage of the job and could be represented as,

$$onJSLookup_E = \left[Stack(Log), \left(\frac{Parallelized * O(n)Storage}{SortByKey(Path)}\right)\right] \quad (4)$$

The '*onBGLookup*' technique returns a synchronous presentation of data blocks generated according to the given stack of logs and could be represented as,

$$onBGLookup_E = \left[Sync\big(Stack(Log)\big), (RDD\ Data)\right] \quad (5)$$

The '*onSCLookup*' method commits the collection of wrapped '*onJSLookup*', '*onBGLookup*' functions and returns control back to the Executor paradigm for performing data block placement strategy in the distributed computing environment.

IV. EXPERIMENTATION

In order to evaluate the proposed approach DCRM, a cluster configuration is adopted to simulate the workflow of Dynamic resource management in Apache Spark.

TABLE 1. CLUSTER CONFIGURATION

Machine	Specifications	No. of VM	
Intel Xeon E5-2600 V2	8 CPUs, 32GM Memory, 1T Disk and 256 GB SDD	3	1 Master Node, 2 Worker Nodes
Intel Core i5	4 Core, 16GM Memory, 1T Disk and 128 GB SSD	2	2 Worker Nodes
Hadoop (Spark)	Spark 2.3.0 (Stable)		
Virtual Machine Management	Virtual Box 5.2		

A. Environment

DCRM is simulated onto a cluster configuration that is evenly divided into two set of computation environment. The first set includes, Intel Xeon processor having 8 CPUs computation power along with a RAM memory of 32GM and persistent storage medias such as 1T Disk and 256 GB SSD. The second set consists of Intel Core i5 processor having 4 Cores along with a RAM memory of 16GB and persistent storage medias i.e., 1T Disk and 128 GB SSD.

B. Experimental Dataset

a) Parallelized Collection Dataset having size of 10 MB
b) External Dataset having size of 10 MB (Food Program) [15]
c) Re-generation of RDD from RDD (a) having size 10 MB

C. Experimental Results

The experiments conducted to evaluate proposed approach includes, (i) RDD Coupling performance and (ii) Job Profiler Workflow.

C.1RDD Coupling Performance

The execution of RDD dataset with an enhancement of DCRM method shows a functional coupling of dynamic resource allocation to the attached types, such as parallelized, external and re-generated datasets. Prior using the coupling identifier, RDD had a limited choice to adopt persistent storage according to ShuffleMapStage, however with an enhancement of coupling identifier, the RDD requests run-time existing persistent storage types and could make a choice according to their respectively availability. Figure-4 shows a flexible enclosure of RDD coupling the processed datasets in the DCRM execution environment of Spark cluster.

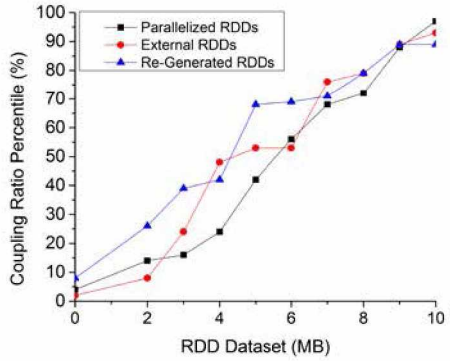


Figure 4. RDD Coupling Performance

C.2Job Profiler Workflow

With the enclosure of coupling attribute in the persistent storage processing, the job is capable of choosing run-time persistent storage according to the preferred ResultStage block processing. After getting the stack logs, we compare the job profile entries to existing pattern of RDD dataset execution and finds that the DCRM synchronize the stack log entries with chosen persistent storage logs and data blocks are in a stable position to be stored in their dynamic preferred persistent storage as shown in Figure-5.

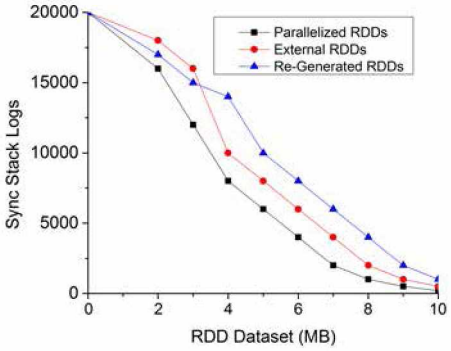


Figure 5. Spark Job Profile Processing

V. CONCLUSIONS

This paper addresses an important operational issue of Apache Spark, that handles storage of Spark job execution in the distributed computing environment. The proposed approach reduces the post ShuffleMapStage stress at cluster and processes data blocks according to dynamically preferred available containers in the Spark Cluster. The experimental evaluations depict that DCRM functionally resolves tight coupling issue from a Spark Job.

In the future, we intend to explore distributed storage coupling capabilities of a Spark Job.

ACKNOWLEDGMENT

This paper was supported by Samsung Research Fund, Sungkyunkwan University, 2018.

REFERENCES

[1] Zaharia, Matei, et al. "Apache spark: a unified engine for big data processing." *Communications of the ACM* 59.11 (2016): 56-65.

[2] Shanahan, James G., and Laing Dai. "Large scale distributed data science using apache spark." *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 2015.

[3] Vavilapalli, Vinod Kumar, et al. "Apache hadoop yarn: Yet another resource negotiator." *Proceedings of the 4th annual Symposium on Cloud Computing*. ACM, 2013.

[4] Salloum, Salman, et al. "Big data analytics on Apache Spark." *International Journal of Data Science and Analytics* 1.3-4 (2016): 145-164.

[5] Zaharia, Matei, et al. "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing." *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*. USENIX Association, 2012.

[6] Choi, I. Stephen, Weiqing Yang, and Yang-Suk Kee. "Early experience with optimizing I/O performance using high-performance SSDs for in-memory cluster computing." *Big Data (Big Data), 2015 IEEE International Conference on*. IEEE, 2015.

[7] Han, Zhijie, and Yujie Zhang. "Spark: A big data processing platform based on memory computing." *Parallel Architectures, Algorithms and Programming (PAAP), 2015 Seventh International Symposium on*. IEEE, 2015.

[8] Agrawal, Dakshi, et al. "SparkBench–a spark performance testing suite." *Technology Conference on Performance Evaluation and Benchmarking*. Springer, Cham, 2015.

[9] Hindman, Benjamin, et al. "Mesos: A Platform for Fine-Grained Resource Sharing in the Data Center." *NSDI*. Vol. 11. No. 2011. 2011.

[10] Krish, K. R., Ali Anwar, and Ali R. Butt. "hats: A heterogeneity-aware tiered storage for hadoop." *Cluster, Cloud and Grid Computing (CCGrid), 2014 14th IEEE/ACM International Symposium on*. IEEE, 2014.

[11] Qureshi, Faseeh, Nawab Muhammad, and Dong Ryeol Shin. "RDP: A storage-tier-aware Robust Data Placement strategy for Hadoop in a Cloud-based Heterogeneous Environment." *KSII Transactions on Internet & Information Systems* 10.9 (2016).

[12] Islam, Nusrat Sharmin, et al. "Efficient data access strategies for Hadoop and Spark on HPC cluster with heterogeneous storage." *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 2016.

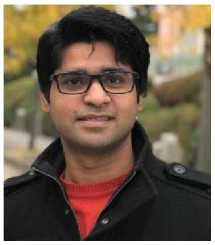
[13] Krish, K. R., et al. "On efficient hierarchical storage for big data processing." *Cluster, Cloud and Grid Computing (CCGrid), 2016 16th IEEE/ACM International Symposium on*. IEEE, 2016.

[14] Serrano, Estefania, et al. "Architecture for the execution of tasks in apache spark in heterogeneous environments." *European Conference on Parallel Processing*. Springer, Cham, 2016.

[15] Hackspark.github.io. (2018). *Download Sample Data - Hack Spark*. [online] Available at: http://hackspark.github.io/environment/download-sample-data/ [Accessed 19 Oct. 2018].

NAWAB MUHAMMAD FASEEH QURESHI (IEEE M'18) is an Assistant Professor in Department of Computer Edu cation, Sungkyunkwan University, Seoul, South Korea. He received Ph.D. degree in Computer Engineering from Sungkyunkwan University, South Korea with the funding support of SAMSUNG scholarship and was awarded with the Superior Research Award from the College of Information and Communication Engineering on account of his research contributions and performance during Ph.D. He is part of editorial and reviewer of various prestigious journals Future Generation Computer Systems, Transactions on Emerging Telecommunications Technologies (ETT), Wireless Personal Communications, KSII Transactions on Internet and Information Systems, Journal of Supercomputing, IEEE Communications Magazine, IEEE Transactions on Industrial Informatics and IEEE Access. He is also reviewer of various top-tier conferences such as IEEE Globecom2018 and IEEE PIMRC 2017. He has been in TCP with IWWCN2017, CSA2017, IMTIC18 and WCSN2017 conferences and performed as session chairs with ICGCET Denmark and soon with RTCSE19 USA. He has facilitated institutes with Webinars on Big data analysis (SZABIST Larkana) and Modern Technology Convergence (MSF South Korea). He was also invited for a keynote talk on convergence with modern technologies in IEEE Pakistan Seminar conduced in MUET. He is an active member of IEEE, ACM, KSII and IEICE. His research interests include big data analytics, context-aware data processing of the Internet of Things, and cloud computing.

ISMA FARAH SIDDIQUI (IEEE M'18) is an Assistant Professor in Department of Software Engineering at Mehran University of Engineering and Technology. She received Ph.D. In Computer Science and Engineering from Hanyang University, South Korea. She was awarded with HEC scholarship for Ph.D. Studies and remained an outstanding candidate with CGPA 4 out of 4 and received Best Thesis Award from the school of Science and Engineering, Hanyang University. She has been an active TPC member of various conferences and is a reviewer of various prestigious journals such as IEEE Access, Future Generation Computer Systems, Journal of Super Computing, IEEE Communications Magazine, KSII Transactions on Internet and Information Systems. Her research interests include semantic web and semantic data analytics, Big Data analytics, context-aware data processing of Internet of Things and green cloud computing.



ASAD ABBAS (IEEE S'17) is an Assistant Professor in Department of Software Engineering, University of Lahore, Lahore Pakistan. He received his BS (Information Technology) degree from "University of the Punjab" Pakistan in 2011. He Joined MS-leading to PhD program in Department of Computer Science and Engineering in Hanyang University ERICA campus at Ansan, South Korea, funded by Higher Education Commission of Pakistan in 2014. He received his Ph.D. degree in Computer Science and Engineering from Hanyang University South Korea in August 2018. His research interests include Software Product Line, Software Requirement Traceability and IoT Applications.

ALI KASHIF BASHIR (IEEE: M'15, SM'16) is working as an Associate Professor in Faculty of Science and Technology, University of the Faroe Islands, Faroe Islands, Denmark. He has published over 60 research articles, 3 white papers, technical reports and several blogs. He received his Ph.D. degree in computer science and engineering from Korea University, South Korea. In the past, he held appointments with Osaka University, Japan; Nara National College of Technology, Japan; the National Fusion Research Institute, South Korea; Southern Power Company Ltd., South Korea, and the Seoul Metropolitan Government, South Korea. He is also attached to Advanced Network Architecture Lab as a joint researcher. He is supervising/co-supervising several graduates (MS and PhD) students. He is in active collaboration with several universities like Yale University and University of Central Florida, USA; Ryerson University, Canada; Osaka University, Japan; to name a few. His research interests include: cloud computing, NFV/SDN, network virtualization, network security, IoT, computer networks, RFID, sensor networks, wireless networks, and distributed computing. He is serving as the Editor-in-chief of the IEEE INTERNET TECHNOLOGY POLICY NEWSLETTER and the IEEE FUTURE DIRECTIONS NEWSLETTER. He is an Editorial Board Member of journals, such as the IEEE ACCESS, the Journal of Sensor Networks, and the Data Communications. He has also served/serving as guest editor on several special issues in journals of IEEE, Elsevier, and Springer. He is actively involved in organizing workshops and conferences in many countries. He has chaired several conference sessions, gave several invited and keynote talks, and reviewed the technology leading articles for journals like IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, the IEEE Communication Magazine, the IEEE COMMUNICATION LETTERS, IEEE Internet of Things, and the IEICE Journals, and conferences, such as the IEEE Infocom, the IEEE ICC, the IEEE Globecom, and the IEEE Cloud of Things. He is also mentoring/advising/counseling several start-ups across the globe in the field of robotics, blockchain and smart homes.

KEE HYUN CHOI is working as a Senior Researcher in Sungkyunkwan University, South Korea. He received his Ph. D from the Sungkyunkwan University and working as Post-Doctoral candidate in the same institute. He has several papers on his account and working in the field of IoT, Big Data and Data Mining.

JAEHYOUN KIM received his B.S. degree in mathematics from Sungkyunkwan University, Seoul, Korea, M.S. degree in computer science from Western Illinois University and Ph.D. degrees in computer science from Illinois Institute of Technology in USA. He was a Chief Technology Officer at Kookmin Bank in Korea before he joined the Department of Computer Education at Sungkyunkwan University in March 2002. Currently he is a professor at Sungkyunkwan University. His research interests include software engineering and architecture, e-Learning, SNS and communication, internet business related policy and computer-based learning.

DONG RYEOL SHIN received the B.S. degree in electrical engineering from the Sungkyunkwan University in 1980, the M.S. degree in electrical engineering from the Korea Advanced Institute of Science and Technology in 1982, and the Ph.D. degree in electrical engineering from the Georgia Institute of Technology, USA, in 1992. From 1992 to 1994, he was with Samsung Data Systems, South Korea, where he was involved in the research of intelligent transportation systems. Since 1994, he has been with the Department of Computer Science and Engineering, Sungkyunkwan University, where he is currently a Full Professor with the Network Research Group. His current research interests lie in the areas of mobile network, ubiquitous computing, cloud computing, and bioinformatics. He is actively involved in the security of vehicular area networks, and the implementation and analysis of Big Data platform, applicable to 3-D image processing of robotic arms.